Quantifying Effectiveness of Activity

Demudu Naganaidu

July 9, 2016

INTRODUCTION

Using devices such as Jawbone Up, Nike FuelBand, and Fitbit it is now possible to collect a large amount of data about personal activity relatively inexpensively. These type of devices are part of the quantified self movement - a group of enthusiasts who take measurements about themselves regularly to improve their health, to find patterns in their behavior, or because they are tech geeks. One thing that people regularly do is quantify how much of a particular activity they do, but they rarely quantify how well they do it.

In this project, I will use data from accelerometers on the belt, forearm, arm, and dumbell of 6 participants. They were asked to perform barbell lifts correctly and incorrectly in 5 different ways.

More information is available from the website here: http://groupware.les.inf.puc-rio.br/har (see the section on the Weight Lifting Exercise Dataset).

DATA

The data for this project come from : http://groupware.les.inf.puc-rio.br/har.

Explore the Data sets

```
dim(training)
```

```
## [1] 19622   160
```

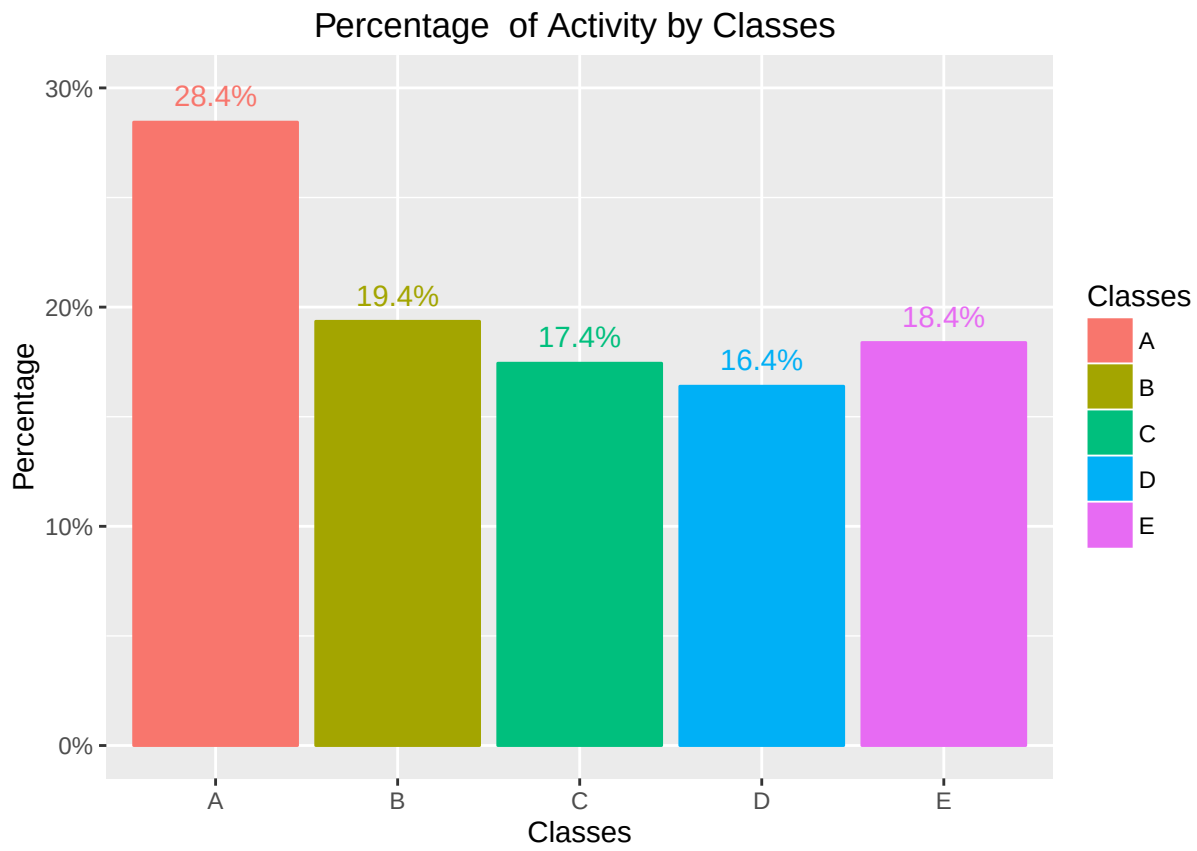
```
dim(testing)
```

```
## [1]  20 160
```

Training data consist of 19,622 and Testing data consist of 20 observations with each 160 variables

Barplot of Classes of Activity by percentage in training data set

```
training.plot
```



Pre-Processing

Pre-Process Training Data

Training data pre-procesed before building model using cleandata function created in the R Script.

Model building

Since training set data is very large with 19,622 observations, we split to training and validation set

I sample 30% of the data for training model and 5% data for validation

```
train <- cleaneddata[sample(nrow(cleaneddata), round(0.3*(dim(training)[1]))), ]
validation <- cleaneddata[sample(nrow(cleaneddata), round(0.05*(dim(training)[1]))), ]
```

The variable to be predicted consist of 5 classess. Thus this a classification problem, as such I propose 4 models that handle classification prediction. Model with best aacuracy will used for prediction with testing data.

Model 1 - using decision tree

```
#Model1 <- train(classe~., method="rpart",data=train)
```

Model 2 - use bagging

```
#Model2 <- train(classe~., method="treebag",data=train)
```

Model 3 - use Random Forest

```
#Model3 <- train(classe~., method="rf",data=train)
```

Model 4 - use Boosting

```
#Model4 <- train(classe~., method="gbm",data=train)
```

EVALUATION OF MODELS

The outcome variable, classe is a categorical variable. Thus Model out of sample error is measured using accuracy from confussion matrix.

1. Decision Tree Model

```
model1sum
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   A   B   C   D   E
##          A 108   3   7   0   0
##          B  43  30  18   0   0
##          C  43   2  50   0   0
##          D  51  13  25   0   0
##          E  19  13  29   0  46
##
## Overall Statistics
##
##                Accuracy : 0.468
##                  95% CI : (0.4236, 0.5128)
##     No Information Rate : 0.528
##     P-Value [Acc > NIR] : 0.9968
##
##                   Kappa : 0.3218
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.4091   0.4918   0.3876       NA   1.0000
## Specificity            0.9576   0.8610   0.8787    0.822   0.8656
## Pos Pred Value         0.9153   0.3297   0.5263       NA   0.4299
## Neg Pred Value         0.5916   0.9242   0.8049       NA   1.0000
## Prevalence             0.5280   0.1220   0.2580    0.000   0.0920
```

```
## Detection Rate         0.2160   0.0600   0.1000    0.000   0.0920
## Detection Prevalence   0.2360   0.1820   0.1900    0.178   0.2140
## Balanced Accuracy      0.6834   0.6764   0.6332       NA   0.9328
```

2. Bagging Model

```
model2sum
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   A   B   C   D   E
##          A 538   5   2   0   1
##          B   7 349   5   2   0
##          C   1   8 346   7   0
##          D   3   2   7 331   2
##          E   0   2   0   2 342
##
## Overall Statistics
##
##                Accuracy : 0.9715
##                  95% CI : (0.9631, 0.9784)
##     No Information Rate : 0.2798
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.964
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9800   0.9536   0.9611   0.9678   0.9913
## Specificity            0.9943   0.9912   0.9900   0.9914   0.9975
## Pos Pred Value         0.9853   0.9614   0.9558   0.9594   0.9884
## Neg Pred Value         0.9922   0.9894   0.9913   0.9932   0.9981
## Prevalence             0.2798   0.1865   0.1835   0.1743   0.1758
## Detection Rate         0.2742   0.1779   0.1764   0.1687   0.1743
## Detection Prevalence   0.2783   0.1850   0.1845   0.1758   0.1764
## Balanced Accuracy      0.9872   0.9724   0.9756   0.9796   0.9944
```

3. Random Forest Model

```
model3sum
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   A   B   C   D   E
##          A 543   2   1   0   0
##          B   3 357   3   0   0
```

```
##          C   0   2 358   2   0
##          D   0   0  11 333   1
##          E   0   0   0   0 346
##
## Overall Statistics
##
##                Accuracy : 0.9873
##                  95% CI : (0.9812, 0.9917)
##     No Information Rate : 0.2783
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9839
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9945   0.9889   0.9598   0.9940   0.9971
## Specificity            0.9979   0.9963   0.9975   0.9926   1.0000
## Pos Pred Value         0.9945   0.9835   0.9890   0.9652   1.0000
## Neg Pred Value         0.9979   0.9975   0.9906   0.9988   0.9994
## Prevalence             0.2783   0.1840   0.1901   0.1707   0.1769
## Detection Rate         0.2768   0.1820   0.1825   0.1697   0.1764
## Detection Prevalence   0.2783   0.1850   0.1845   0.1758   0.1764
## Balanced Accuracy      0.9962   0.9926   0.9786   0.9933   0.9986
```

4. Gradient Boosting Model

```
model4sum
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   A   B   C   D   E
##          A 117   0   1   0   0
##          B   5  81   3   0   2
##          C   1   6  85   3   0
##          D   0   0   2  87   0
##          E   0   3   0   3 101
##
## Overall Statistics
##
##                Accuracy : 0.942
##                  95% CI : (0.9178, 0.9608)
##     No Information Rate : 0.246
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9273
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
```

```
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9512   0.9000   0.9341   0.9355   0.9806
## Specificity            0.9973   0.9756   0.9756   0.9951   0.9849
## Pos Pred Value         0.9915   0.8901   0.8947   0.9775   0.9439
## Neg Pred Value         0.9843   0.9780   0.9852   0.9854   0.9949
## Prevalence             0.2460   0.1800   0.1820   0.1860   0.2060
## Detection Rate         0.2340   0.1620   0.1700   0.1740   0.2020
## Detection Prevalence   0.2360   0.1820   0.1900   0.1780   0.2140
## Balanced Accuracy      0.9743   0.9378   0.9548   0.9653   0.9827
```

ACCURACY

accuracy_table <- data.frame(c("Model 1", "Model 2", " Model 3"," Model 4"), c(model1sum*overall*[1], *model*2*sum*overall[1],mo
names(accuracy_table) <- c("Model","Accuracy")

Based on the accuracy of the above 4 confusionmatrix. The summary of the accuracy is as follows:

```
accuracy_table
```

```
##      Model  Accuracy
## 1  Model 1 0.4680000
## 2  Model 2 0.9714577
## 3  Model 3 0.9872579
## 4  Model 4 0.9420000
```

Both Model 2 and Model 3 perform better than Model 1 and Model 4 . Between Model 2 and Model 3, Model 3 perform slightly better. As such Model 3 choosed to predict the validation data set with 20 observations.

Prediction on the Testing Data set

Clean the testing data set

```
#testing_data<- cleandata(testing)
```

Get the predictions

```
#testing_predict <- predict(Model3,newdata = testing_data)
```

My final predictions are :

```
testing_predict
```

```
##  [1] B A B A A E D B A A B C B A E E A B B B
## Levels: A B C D E
```

Note: The detail code for this project is found in the github. Please click Quantifying Effectiveness of Activity.R in the Github.

REFERENCE

The above study and data for this project was genourously shared by:

Velloso, E.; Bulling, A.; Gellersen, H.; Ugulino, W.; Fuks, H. Qualitative Activity Recognition of Weight Lifting Exercises. Proceedings of 4th International Conference in Cooperation with SIGCHI (Augmented Human '13) . Stuttgart, Germany: ACM SIGCHI, 2013.

More information is available from the website here: http://groupware.les.inf.puc-rio.br/har (see the section on the Weight Lifting Exercise Dataset).